Гладка Л. І.
кандидат фізико-математичних наук,
доцент кафедри автоматизації та
комп'ютерно-інтегрованих технологій
Черкаського національного університету
імені Богдана Хмельницького, Черкаси, Україна
Бодненко Т. В.
кандидат фізико-математичних наук,
доцент кафедри автоматизації та
комп'ютерно-інтегрованих технологій
Черкаського національного університету
імені Богдана Хмельницького, Черкаси, Україна

АДАПТАЦІЯ МІЖНАРОДНИХ СТАНДАРТІВ НАВЧАННЯ ПРОГРАМУВАННЮ У ВИЩИХ НАВЧАЛЬНИХ ЗАКЛАДАХ ДО НАЦІОНАЛЬНОЇ СИСТЕМИ ОСВІТИ

Нині склалася парадоксальна ситуація, характерна для багатьох розвинених країн, в тому числі України. Незважаючи на те, що роль інформаційних технологій швидко зростає в усіх сферах діяльності, попит на IT-фахівців, які і так мають відносно високі зарплати теж зростає, тому актуальним залишається

питання підготовки спеціалістів ІТ галузей. При цьому навіть збільшення кількості бюджетних місць на ІТ напрями, не ліквідовує існуючі проблеми:

1. Випуск ІТ фахівців істотно відстає від поточних потреб ІТ бізнесу за кількістю, а також не збігається за структурою та переліком пропозиції на ринку праці в ІТ галузі.
2. Існує проблема якості випуску, за статистикою приблизно лише кожен 4-й випускник ІТ спеціальності влаштовується працювати за спеціальністю, що є дуже низьким показником. Причина в тому, що державна підготовка ІТ фахівців розвивається без зв'язку з ІТ галуззю [1].
3. Недостатня кількість і якість випускників середніх шкіл зумовлюють низький рівень абітурієнтів та обмежують кількість і якість випущених ІТ фахівців [2].

Отже, якісна підготовка фахівців ІТ галузей є важливою освітньою задачею нашої держави.

Метою статті є адаптація міжнародних методологічних підходів навчання програмуванню у вищих навчальних закладах до національної системи освіти та розробка схеми, яка містить методологію програмування, технологічний підхід, а також послідовність мов програмування, які важливо вивчати майбутньому ІТ фахівцю.

Дослідженнями проблем підготовки інженерів-програмістів та актуальними питаннями комп'ютингу займалися в галузі професійної підготовки: А.Т. Ашеров [3], М.І. Шкіль і ін.; роботи в галузі методології інформатики: В.Ю. Биков [4], В.М. Глушков, А.М. Гуржій, М.І. Жалдак і ін.; в галузі методики навчання інформатики: А.Ф. Верлань, М.І. Жалдак [5], О.В. Співаковський, Ю.В. Триус і ін.; в галузі актуальних питань підготовки програмістів: Бертран Мейер [6], П. Денінг, Д. Кнут, С.О. Семеріков [7,8], Т.Ю. Морозова [9] і ін.

При дослідженні описаних вище проблеми, науковці [6, 7] стверджують, що основною причиною вказаних явищ є складність університетських навчальних програм та їх абстрактність, недостатній зв'язок і слабка кореляція між практичними потребами і надзвичайно швидким розвитком ІТ-галузі.

Вивчення програмування спрямоване на засвоєння студентами основних концепцій програмування і на формування первинних практичних навичок. До останнього десятиліття більшість університетів будували свою стратегію викладання відповідно до рекомендацій Association for Computing Machinery (ACM) Computing Curriculum (2016) [10]. Ця програма заснована на математичній методиці викладання дисциплін програмування (Дейкстра Е.В., 1997) [11], оскільки припускалося, що програмування повинно розглядатися як предметна область математики. У Association for Computing Machinery (ACM) Computing Curriculum (2016) була представлена розширена структура знань основ програмування, огляд моделей викладання матеріалу [10], де тільки половина навчальної програми, присвячена алгоритмам опису та аналізу даних, в той час як друга половина присвячена різним питанням програмної інженерії. У той же час, у вказаному документі університетам було рекомендовано розробити оригінальні навчальні програми з урахуванням Computing Curriculum.

Постійне оновлення апаратного та програмного забезпечення, поява нових ґаджетів та їх платформ, впливає на зміст курсів навчальних програм підготовки фахівців ІТ-галузі. За словами професора Клауса Шваба, людство стоїть на порозі нової технічної революції – цифрової. Третя промислова революція -

автоматизувала виробництво за допомогою електроніки та інформаційних технологій. Четверта промислова революція спирається на третю - з середини минулого століття триває цифрова революція в усіх сферах життя [12].

З точки зору навчання програмуванню, будемо розглядати програмування як науку, для вивчення якої потрібна своя педагогічна система. Хоча класики програмування неоднозначні у своїх поглядах на вказане питання. Одні вважають програмування наукою, інші мистецтвом, треті - майстерністю. Наведемо цитати, автор кожної з яких висловлює своє ставлення до цієї проблеми.

Дональд Кнут в монографії "The Art of Computer Programming" [13] зробив порівняльний аналіз програмування і мистецтва: "... computer programming is an art, because it applies accumulated knowledge to the world, because it requires skill and ingenuity, and especially because it produces objects of beauty. A programmer who subconsciously views himself as an artist will enjoy what he does and will do it better" (в оригіналі).

Девід Гріс (David Gries) у передмові до монографії [14] розпочинає з наступного твердження: " Програмування розпочиналося як мистецтво, і навіть сьогодні більшість людей навчаються лише спостерігаючи за іншими виконавцями (наприклад, лектором, другом) та через звичку, осягаючи прийоми і мало замислюючись над принципами, які лежать в їх основі. Однак в результаті наукових досліджень останнього десятиліття знайдені деякі корисні теоретичні положення і загальні принципи, тому настає час, коли можна починати навчати принципам і їх усвідомленому застосуванню. Дана робота є спробою передати моє розуміння і захоплення програмуванням як вищою наукою." (в оригіналі "Programming began as an art, and even today most people learn only by watching others perform (e.g. a lecturer, a friend) and through habit, with little direction as to the principles involved. In the past 10 years, however, research has uncovered some useful theory and principles, and we are reaching the point where we can begin to teach the so that they can be consciously applied. This text is an attempt to convey my understanding of and excitement for this just-emerging science of programming.").

У той же час перші слова книги Уезерелла "Етюди для програмістів" [15] звучать так: "Програмування - це майстерність, і кожен програміст повинен досягти потрібного професійного рівня".

Сформулюємо основну задачу професійного програмування як розробку якісного програмного продукту. Перш ніж розглядати методологію вивчення програмування, визначимо головну ціль навчання програмуванню як формування матриці компетенцій програміста [17]. Для досягнення цілі необхідно дотримуватись принципів навчання, в основі яких лежать закони та закономірності дидактичного процесу. Короткі рекомендації по їх застосуванню виглядають так:

Перший етап. Спочатку слід визначити методологію програмування, яка буде включати сукупність методів і концепцій, об'єднаних загальним філософським підходом.

Коли методологія застосовується у програмуванні, часто її називають парадигмою програмування, тобто способом мислення, не пов'язаним з конкретним мовою програмування.

Ядра методологій визначаються механізмом опису алгоритмів. Перелічимо основні ядра методологій, які важливо засвоїти у процесі навчання фахівцям ІТ

напрямів підготовки:

1. методологія структурного програмування;
2. методологія імперативного програмування;
3. методологія об'єктно-орієнтованого програмування;
4. методологія функціонального програмування;
5. методологія логічного програмування.

Другий етап. Далі слід вибрати технологічний підхід, який буде визначати сукупність процесів, які використовуються при розробці програмного продукту. Визначена раніше методологія включає сукупність методів, які будуть застосовані в технологічному підході.

Слід розрізняти методи програмування від методів навчання програмуванню. Існує дев'ять ідентифікованих методів, використовуваних при навчанні програмуванню.

1. Опитування (Survey).
2. Проектування програмного забезпечення і реалізація (Software design and development).
3. Метод тестувань і перевірок (Experience report).
4. Метод порівняльних дослідженнь (Comparison study).
5. Мета-аналіз / огляд літератури (Meta Analysis/ Literature review).
6. Оцінювання програми (Program evaluation).
7. Метод емпіричного дослідження (Empirical study).
8. Розробка моделі та структури (Model and framework development).
9. Case study (відсутній точний переклад на українську мову). Методика ситуативного навчання, заснована на вивченні реального об’єкта з його предметною областю, проектування рішення у запропонованій ситуації. Включає в себе опис конкретної практичної ситуації з постановкою проблеми, довідкову і додаткову інформацію, методичні матеріали та вказівки.

Третій етап. Методологія і технологія визначають мови і системи програмування, необхідні для кожного процесу обраного технологічного підходу.

Ми визначимо важливі умови вибору мов програмування та побудови послідовності вивчення мов програмування.

1. Характеристика й особливості мови програмування (призначення; тип; особливості синтаксису).

2. Наявність вільного, зручного для використання в навчанні середовища програмування.

3. Наявність методичної підтримки (інформаційно-дидактичного, навчально-методичного забезпечення).

Зрозуміло, що будь-яка мова програмування намагається задовольнити певній сукупності вимог, нехтуючи іншими.

Мови програмування поділяються на типи за їх парадигмою. Більш поширеними у навчанні були процедурні, об’єктні, візуальні мови програмування. Нині актуальним є мультипарадигмальне програмування, оскільки мультипарадигмальна мова програмування – це мова, яка підтримує більше ніж одну парадигму програмування, наприклад: процедурне, об’єктно-орієнтоване, функціональне та інші програмування.

На рис. 1. представлено розроблена нами схема, яка містить методологію програмування, технологічний підхід, а також послідовність мов програмування,

які, на нашу думку, важливо вивчати майбутньому ІТ фахівцю.

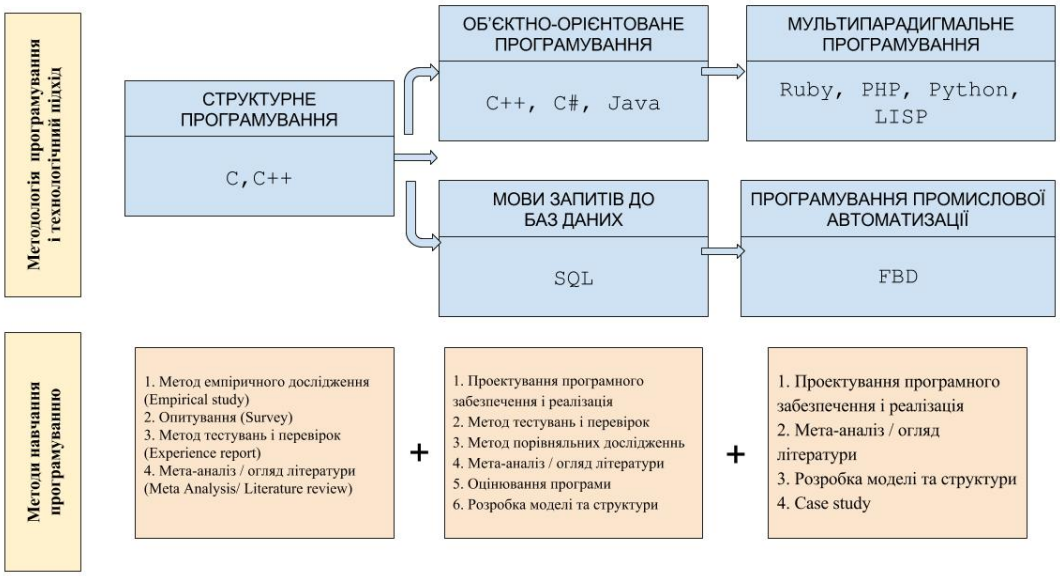

Рис. 1. Методологія програмування, технологічний підхід та методи навчання програмуванню

Четвертий етап. Технологічні процеси будуть виконуватися на деяких апаратній та операційній платформах. Зауважимо, що апаратна і операційна платформи можуть істотно визначати наявність і специфіку інструментів. У більшості розробок слід уникати залежності від платформ.

Інструменти, що використовуються тьюторами для надання допомоги студентам при вивченні програмуванні, включають в себе візуалізацію, симуляцію, фізичні або онлайн інструменти.

Окрім інструменту програмування, важливо також використовувати інструмент управління або підтримки, особливо для оцінки роботи студентів, для планування заходів та спілкування.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Дослідження ефективності ІТ освіти [Електронний ресурс]. – Режим доступу: http://itukraine.org.ua/doslidzhennya. – Назва з екрана.
2. Інформаційно-пошукова система «Конкурс» [Електронний ресурс]. – Режим доступу: http://www.vstup.info. – Назва з екрана.
3. Ашеров А.Т. Управление качеством учебно-познавательной деятельности студентов при компьютерном обучении. [Ч. 2. Стратегии обучения и контроля] / А.Т. Ашеров // Проблеми інж.-пед. освіти. - 2007. - № 16. - С. 16-26.
4. Биков В.Ю. Моделі організаційних систем відкритої освіти / В.Ю. Биков. // Київ: Атіка, 2009. – 684 с.
5. Жалдак М.І. Використання комп'ютера в навчальному процесі має бути педагогічно виваженим і доцільним / М.І. Жалдак // Комп'ютер в школі та сім'ї. – 2011. – № 3 – С. 3-12.
6. Meyer B. The Outside-In Method of Teaching Introductory Programming. [Electronic Resource]. – Mode of access: URL: http://se.ethz.ch/~meyer/publications/teaching/teaching-ispj.pdf. – Title from the screen.
7. Семеріков С.О. Фундаменталізація навчання інформатичних дисциплін у вищій школі: Монографія / Наук. ред. М.І. Жалдак. — Кривий Ріг: Мінерал; К.: НПУ ім. М.П. Драгоманова, 2009. — С. 55–56.
8. Семеріков С.О. Стабілізація курсів інформатики як засіб фундаменталізації інформатичних дисциплін / С. О. Семеріков // Рідна школа. - 2008. - №5. - С. 11-12.
9. Морозова Т.Ю. Взаємозв'язок освітніх програм ІТ профіля та ІТ професій (з міжнародного досвіду) [Електронний ресурс] / Морозова Т.Ю. — Режим доступу: http://old.apitu.org.ua/node/503. – Назва з екрана.

10. Computer Science 2016: Curriculum Guidelines for Undergraduate Programs in Computer Science [Electronic Resource]. – Mode of access: URL: https://www.computer.org/cms/Computer.org/professional-education/curricula/ComputerEngineeringCurricula2016.pdf. – Title from the screen.
11. Edsger Wybe Dijkstra. A Discipline of Programming / Prentice Hall. – 1976.- 217pp.
12. Klaus Schwabю The Fourth Industrial Revolution [Electronic Resource]. – Mode of access: URL: https://www.foreignaffairs.com/articles/2015-12-12/fourth-industrial-revolution. – Title from the screen.
13. ACM Turing award lectures*:* the first twenty years*:* 1966-1985 / ACM Press/Addison-Wesley Publishing Co. New York, NY, USA. - 1987.
14. David Gries. The Science of Programming. Texts and Monographs in Computer Science / David Gries // Springer-Verlag, 1981.
15. Charles Wetherell. Etudes for Programmers / Charles Wetherell // Prentice Hall, 1978. – 200 p.
16. Цейтин Г.С. О профессионализме в программировании / Г.С. Цейтин // СПб.: ЛГУ, математико-механический факультет, рукопись, 1989.
17. Programmer Competency Matrix [Electronic Resource]. – Mode of access: URL: https://software-carpentry.org/blog/2010/12/programmer-competency-matrix.html – Title from the screen.